**1. Part 1**

Using a time stamp counter to calculate clock cycles used to be pretty reliable and with low overhead. However, with new processor abilities such as multi-threading, there are new challenges that arise. Some challenges that face the time stamp counter are: First, new processor have multiple cores or multiple CPUs. There is no guarantee that the the time stamp counter in different CPUs will be synchronized. Second, new features like hibernation makes the count inaccurate. Even if the process runs on only one CPU, there is still the issue of the OS or BIOS putting the CPU in hibernating mode. This would throw off the counter value. Third, starting with the Pentium Pro, Intel processors use out-of-order execution. This allows for the instructions to be executed in a different order than in the program code. This could cause the counter value to be off as well. Fourth, if the measured time is too long, there could be a problem of overflowing the value.

**2. Part 2**

To handle the issue of having too small of a measurement interval, the source code loops through and calls the function until the minimum threshold is met. To handle the issue of repeatability, the source code calls the function multiple times and stores the best time in an array.

**3. Part 3**

In vector.h there are several parameters that can be changed. “data\_t” is the data type that the operations will be on. “OP” is the kind of operation to perform. In the Makefile there are flags that can be set that changes the optimization level when compiling with gcc. In “getcpe.c”, “VECVALS” changes the number of unique vector lengths tested. “VECMAX” changes the max vector length. “MEASMAX” changes the number of runs to make each vector length.

The two parameters I was checking was “VECVALS” and “VECMAX”. I found that to match the book's results for integer addition closest, it is best to set them to 1000 and 1024 respetively. So “VECVALS” had to be increased while “VECMAX” stayed the same.

**4. Part 4**

**5.**

Comparing the values between what the book had and what I received, I noticed that the trend is pretty much the same. The difference between the values is that mine are usually a little lower. There could be lots of causes that would create this difference. One difference could be this version of gcc is just better at optimizing that the version of gcc that the textbook uses.